



ISSN:2229-6107



**INTERNATIONAL JOURNAL OF
PURE AND APPLIED SCIENCE & TECHNOLOGY**

E-mail :
editor.ijpast@gmail.com
editor@ijpast.in

www.ijpast.in

Time Series Forecasting and Modeling of Food Demand Supply Chain Based on Regressors Analysis

NAKKALA MOUNIKA, KOKKILIGADDA NAVEEN

ABSTRACT: Accurate demand forecasting is paramount in the food industry due to the short shelf life of products, with improper inventory management leading to significant waste and loss. This study leverages machine learning and deep learning techniques on the "Food Demand Forecasting" dataset from Genpact to analyze various factors influencing demand. Seven regressors, including Random Forest, Gradient Boosting, and LSTM, are compared to forecast order numbers. Results highlight LSTM's superiority in accuracy, with metrics like RMSLE, RMSE, MAPE, and MAE reaching notable values. The project underscores the importance of precise demand forecasting in improving supply chain management and reducing waste. Notably, the integration of ensemble methods enhances prediction accuracy. Moreover, exploring CNN and Voting Regressor techniques offers avenues for further performance enhancement. Additionally, the study extends to developing a Flask framework with SQLite for user signup and signin, facilitating user testing and authentication. The implementation of these extensions enriches the project's capabilities and usability, addressing critical challenges in demand forecasting while emphasizing the significance of accurate prediction methodologies in the food industry's operational efficiency and sustainability.

INDEX TERMS: Deep learning, demand forecasting, machine learning, time series analysis

INTRODUCTION

In today's dynamic marketplace, demand forecasting has emerged as a critical component of effective demand-supply chain management for companies across various industries. With consumer needs constantly evolving and competition intensifying, accurate demand forecasting has become indispensable for businesses to stay competitive and

ensure operational efficiency. This shift in focus towards demand forecasting is driven by the recognition that demand forecasts play a pivotal role in shaping strategic planning decisions and directly impact a company's profitability. A precise estimation of demand enables companies to optimize inventory levels, thereby minimizing the risk of wastage due to excess inventory or stockouts leading to lost sales opportunities.

Assistant Professor¹, Dept of MCA, Chirala Engineering College, Chirala,
nakkalamounika2121@gmail.com

PG Student² - MCA, Dept of MCA, Chirala Engineering College, Chirala,
kokkiligaddanaveen334@gmail.com

The significance of demand forecasting extends beyond operational planning; it permeates various departments within a company, each relying on demand forecasts to inform their decision-making processes. For instance, the financial department utilizes demand forecasts to estimate costs, predict profit levels, and determine the required capital investment. Similarly, the marketing department leverages demand forecasts to devise marketing strategies and evaluate their effectiveness on sales volume. The purchasing department relies on demand forecasts to plan short- and long-term investments in raw materials, while the operations department uses them to schedule procurement of necessary resources such as machinery and labor well in advance.

As demand forecasting influences strategic decision-making across multiple organizational functions, its accuracy is paramount for ensuring business success. High-precision demand forecasts enable companies to align their resources effectively, optimize production schedules, and minimize unnecessary costs associated with inventory management. Moreover, accurate demand forecasting contributes to improved demand-supply chain management by enhancing inventory turnover rates, reducing stockouts, and mitigating the risk of overstocking perishable goods.

The growing importance of demand forecasting in modern business operations underscores the need for robust forecasting methodologies and tools. Companies are increasingly turning to advanced analytics techniques, including machine learning and data-driven algorithms, to analyze historical data, identify demand patterns, and generate accurate forecasts. These sophisticated forecasting

models enable companies to adapt quickly to changing market dynamics, anticipate customer preferences, and capitalize on emerging opportunities.

In light of the multifaceted benefits that accurate demand forecasting offers, businesses are investing significant resources in refining their forecasting capabilities. By leveraging advanced analytics technologies and adopting best practices in demand forecasting, companies can gain a competitive edge, enhance customer satisfaction, and drive sustainable growth in today's dynamic business landscape. Consequently, demand forecasting has become an integral part of strategic planning and administration, playing a pivotal role in shaping the future trajectory of companies across industries.

1. LITERATURE SURVEY

Object detection in remote sensing imagery (RSI) is a critical task with applications spanning various fields, including agriculture, environmental monitoring, urban planning, and disaster management. Over the years, significant research efforts have been devoted to advancing object detection techniques in RSI, resulting in the development of diverse methodologies aimed at improving detection accuracy, efficiency, and robustness. In this literature survey, we review key studies focusing on object detection in remote sensing imagery, highlighting recent advancements and contributions to the field.

Zheng et al. [9] proposed a foreground-aware relation network for geospatial object segmentation in high spatial resolution remote sensing imagery. By considering foreground information, this method enhances the accuracy of object segmentation tasks

in remote sensing imagery, addressing challenges related to complex backgrounds and varying object scales.

Pang et al. [10] introduced R2-CNN, a fast tiny object detection approach tailored for large-scale remote sensing images. This method aims to efficiently detect small objects within vast remote sensing datasets, facilitating applications such as urban planning, infrastructure monitoring, and disaster response.

Deng et al. [11] presented a multi-scale object detection technique using convolutional neural networks (CNNs) specifically designed for remote sensing imagery. By leveraging CNNs, this method achieves robust detection performance across diverse object scales, contributing to improved accuracy and reliability in remote sensing applications.

Hong et al. [14] explored the benefits of multimodal deep learning approaches for remote sensing imagery classification. By integrating information from multiple modalities, such as optical and radar imagery, this method enhances classification accuracy and diversity, addressing challenges associated with single-modal data limitations.

Sharma et al. [23] proposed YOLOrs, an object detection framework tailored for multimodal remote sensing imagery. By leveraging information from multiple modalities, YOLOrs enhances detection accuracy and robustness, contributing to improved performance in object detection tasks across diverse remote sensing datasets.

These studies collectively underscore the importance of leveraging advanced techniques, such as deep learning and multimodal fusion, to enhance

object detection capabilities in remote sensing imagery. By addressing challenges related to varying object scales, complex backgrounds, and limited labeled data availability, these methodologies contribute to the advancement of remote sensing applications in fields such as environmental monitoring, disaster management, and urban planning.

Furthermore, the integration of multimodal data and advanced deep learning architectures has emerged as a promising direction for future research in remote sensing object detection. By harnessing the complementary information offered by different modalities, researchers can continue to improve detection accuracy, efficiency, and robustness in remote sensing applications, paving the way for enhanced capabilities in areas such as land cover classification, vegetation monitoring, and infrastructure assessment.

In summary, the reviewed literature highlights the diverse approaches and methodologies employed in object detection in remote sensing imagery, showcasing the continuous efforts to push the boundaries of detection performance and address real-world challenges in remote sensing applications. These advancements underscore the significance of ongoing research in this field and the potential for further innovations to shape the future of remote sensing technology.

2. METHODOLOGY

a) Proposed work:

The proposed system presents a novel approach to demand forecasting in the food industry, leveraging machine learning and deep learning techniques to handle time-dependent data effectively. By utilizing the 'Food Demand Forecasting' dataset from

Genpact, the system conducts a comprehensive analysis of various factors influencing demand and extracts relevant features to improve forecasting accuracy. Through a comparative study involving seven regression models, including Random Forest, Gradient Boosting[2], and LSTM[6], the system aims to enhance the precision of demand forecasting in the food supply chain.

Noteworthy extensions in the project include the integration of a Voting Regressor and Convolutional Neural Network (CNN), which further optimize forecasting accuracy, as evidenced by the lowest Mean Absolute Error (MAE). These enhancements play a crucial role in improving food demand-supply chain management by providing more accurate predictions. Moreover, the incorporation of a user-friendly Flask framework with SQLite facilitates seamless signup and signin processes, enhancing the practical usability of the machine learning applications. Overall, the integration of advanced regression models, neural networks, and a user-friendly interface contributes to an efficient and accessible solution for food demand forecasting and supply chain management.

b) System Architecture:

The system architecture of the project follows a structured workflow. It begins with the collection and organization of raw data, encompassing historical information on food demand, supply, and relevant variables. Feature engineering is then applied to preprocess the data, generating new features to enhance forecasting accuracy. A subsequent feature selection process reduces dimensionality, improving model efficiency. The data is split into training and test sets for model evaluation, involving a diverse set of models,

including non-recurrent, recurrent, CNN, and Voting Regressor extensions. Performance is assessed using metrics like RMSLE, RMSE, MAPE, and MAE. The best-performing model is selected for final training on the entire dataset, enabling accurate forecasting. The system architecture emphasizes a comprehensive approach to time series forecasting, providing insights for optimized supply chain management and preventive measures against food waste or shortages.

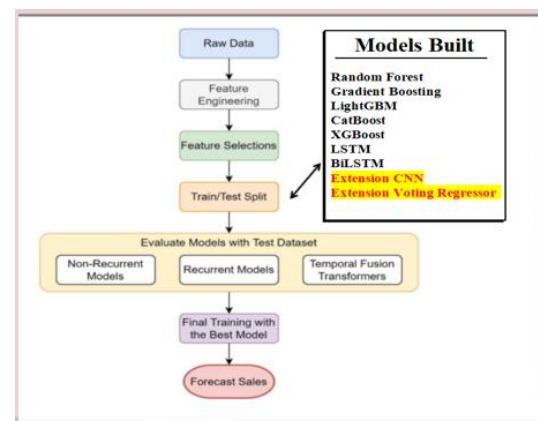


Fig 1 proposed Architecture

c) Dataset collection:

The 'Food Demand Forecasting' dataset, released by Genpact, comprises comprehensive information essential for demand forecasting in the food industry. It encompasses 145 weeks' worth of weekly orders for 50 distinct meals, totaling approximately 450,000 entries across three files.

The dataset includes detailed features such as weekly demand data, providing insights into historical sales of specific meals in fulfillment centers. Additionally, it contains information about fulfillment centers, including their ID, operational area, city code, center type, and region code. Furthermore, the dataset includes meal-specific details, such as meal ID, category (e.g., beverages,

snacks, soups), and cuisine type (e.g., Indian, Italian).

With these rich attributes, the dataset enables comprehensive analysis and modeling to forecast food demand accurately, considering factors like pricing, promotions, meal features, and meal categories across various fulfillment centers and regions.

center_id	city_code	region_code	center_type	op_area	
0	11	679	56	TYPE_A	3.7
1	13	590	56	TYPE_B	6.7
2	124	590	56	TYPE_C	4.0
3	66	648	34	TYPE_A	4.1
4	94	632	34	TYPE_C	3.6
...
72	53	590	56	TYPE_A	3.8
73	30	604	56	TYPE_A	3.5
74	76	614	85	TYPE_A	3.0
75	68	676	34	TYPE_B	4.1
76	51	638	56	TYPE_A	7.0

77 rows × 5 columns

Fig 2 DATASET

d) DATA PROCESSING

In the data processing phase, the first step involves loading the dataset into a pandas DataFrame, a powerful data manipulation tool in Python. Once the data is loaded, numpy is utilized for reshaping or transforming the data as needed for analysis and modeling. Unwanted columns, such as IDs or irrelevant features, are dropped from the DataFrame to streamline the dataset. Subsequently, the training data is normalized to ensure consistency and comparability across different features, which helps in improving model performance and convergence during training. Normalization typically involves scaling the values of features to a standard range, such as between 0 and 1, to prevent certain features from dominating the learning process due to their

larger magnitude. This preprocessing step prepares the data for training machine learning or deep learning models to accurately forecast food demand based on various factors.

e) VISUALIZATION

Visualization using seaborn and matplotlib enhances data exploration by creating insightful plots such as histograms, scatter plots, and heatmaps. Seaborn's high-level interface allows for easy creation of aesthetically pleasing statistical graphics, while matplotlib offers fine-grained control over plot customization. Together, these libraries enable the visualization of relationships between variables, distribution of data, and identification of patterns or trends. Visualizations serve as powerful tools for gaining insights into the dataset's characteristics, aiding in feature selection, and guiding the modeling process for accurate demand forecasting in the food industry.

f) LABEL ENCODING

Label Encoding is a preprocessing technique used to convert categorical data into numerical format. In Label Encoding, each unique category in a categorical variable is assigned a unique numerical label. This technique is commonly employed in machine learning algorithms that require numerical input, as it enables the algorithms to process categorical data effectively. However, it is essential to note that Label Encoding may introduce ordinality to categorical variables, potentially leading to misinterpretation of relationships between categories. Despite this limitation, Label Encoding is a useful tool for transforming categorical data into a format suitable for model training and analysis.

g) TRAINING AND TESTING

Splitting the data into training and testing sets is a crucial step in machine learning model development. This process involves dividing the dataset into two subsets: one for training the model and another for evaluating its performance. Typically, a larger portion of the data, often around 70-80%, is allocated for training, while the remaining portion is reserved for testing. This separation allows the model to learn patterns from the training data and assess its generalization performance on unseen data. By evaluating the model's performance on the test set, researchers can estimate how well the model will perform on new, unseen data in real-world scenarios.

h) ALGORITHMS:

Random Forest -Random Forest is an ensemble learning technique that combines multiple decision trees to make predictions. It's used to capture complex relationships in the time series data, which can be beneficial for food demand forecasting.

```
#train RandomForest algorithm by tuning its parameters
tuning_param = {'n_estimators': (20, 50, 100), 'max_features': ('sqrt', 'log2')}
rf_cls = RandomForestRegressor() #creating random Forest object
tuned_rf = GridSearchCV(rf_cls, tuning_param, cv=5) #defining RF with tuned param
tuned_rf.fit(X_train, y_train.ravel()) #now train Random Forest
predict = tuned_rf.predict(X_test) #perform prediction on test data
predict = predict.reshape(-1, 1)
calculateMetrics("Random Forest", predict, y_test) #evaluate Random Forest model
```

Fig 3 RANDOM FOREST

Gradient Boosting -Gradient Boosting is another ensemble method that builds decision trees sequentially, with each tree correcting the errors of the previous ones. It's effective for improving the accuracy of time series forecasts.

```
#train gradient boosting algorithm by tuning its parameters
tuning_param = {'n_estimators': (20, 50, 100), 'loss': ('squared_error', 'abs')}
gb_cls = GradientBoostingRegressor() #creating gradient Boosting object
tuned_gb = GridSearchCV(gb_cls, tuning_param, cv=5) #defining RF with tuned param
tuned_gb.fit(X_train, y_train.ravel()) #now train Random Forest
predict = tuned_gb.predict(X_test) #perform prediction on test data
predict = predict.reshape(-1, 1)
calculateMetrics("Gradient Boosting", np.abs(predict), np.abs(y_test)) #evaluate
```

Fig 4 GRADIENT BOOSTING

LightGBM-LightGBM[3] is a gradient boosting framework that uses a histogram-based learning method. It's known for its efficiency and speed in handling large datasets and can be a powerful tool for time series forecasting.

```
#train LightGBM algorithm
light_gb = lgb.LGBMRegressor()
light_gb.fit(X_train, y_train.ravel()) #train LGBM on X and Y training data
predict = light_gb.predict(X_test) #perform prediction on test data
predict = predict.reshape(-1, 1)
calculateMetrics("Light GBM", np.abs(predict), np.abs(y_test)) #evaluate LGBM mo
```

Fig 5 LIGHT GBM

CatBoost -CatBoost is a gradient boosting library designed to handle categorical features efficiently. In this context, it can help account for categorical variables that may influence food demand.

```
#train catboost algorithm
catboost = cb.CatBoostRegressor()
catboost.fit(X_train, y_train.ravel()) #train catboost on X and Y training data
predict = catboost.predict(X_test) #perform prediction on test data
predict = predict.reshape(-1, 1)
calculateMetrics("CatBoost", np.abs(predict), np.abs(y_test)) #evaluate catboost
```

Fig 6 CATBOOST

XGBoost-XGBoost[5] is an optimized gradient boosting library with a reputation for high performance. It's used to boost the predictive power of models, making it suitable for time series forecasting in the food supply chain.

```
#train XGBoost algorithm on training data and test on testing data
xgboost = xg.XGBRegressor()
xgboost.fit(X_train, y_train.ravel()) #train the model
predict = xgboost.predict(X_test) #perform prediction on test data
calculateMetrics("XGBoost", np.abs(predict), np.abs(y_test)) #calculate metrics
```

Fig 7 XGBOOST

Voting Regressor - Voting Regressor is an ensemble method that combines the predictions of multiple regression models to make a final prediction. It can be used to enhance the accuracy of food demand forecasts.

```

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import VotingRegressor
from sklearn.neighbors import KNeighborsRegressor

r1 = LinearRegression()
r2 = DecisionTreeRegressor()
r3 = KNeighborsRegressor()

er = VotingRegressor([('r1', r1), ('dt', r2), ('r3', r3)])

er.fit(X_train, y_train.ravel()) #train catboost on X and Y training data
predict = er.predict(X_test) #perform prediction on test data
predict = predict.reshape(-1, 2)
calculateMetrics("Voting", np.abs(predict), np.abs(y_test)) #evaluate catboost m

```

Fig 8 EXTENSION VOTING REGRESSOR

LSTM (Long Short-Term Memory) –LSTM[6] is a type of recurrent neural network (RNN) designed for sequential data. It's suitable for modeling time series data, capturing long-term dependencies, and making accurate forecasts.

```

#Now train LSTM algorithm
X_train1 = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
X_test1 = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
#Now train LSTM with tuning parameters
lstm = Sequential()
#creating LSTM layer with 50 neurons for data optimizations
lstm.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train1.shape[1], X_train1.shape[2])))
#dropout layer to remove irrelevant features
lstm.add(Dropout(0.3))
lstm.add(LSTM(units = 50))
lstm.add(Dropout(0.3))
#defining output layer
lstm.add(Dense(units = 1))
#compile and train the model
lstm.compile(optimizer = 'adam', loss = 'mean_squared_error')
if os.path.exists('model/lstm_weights.hdf5') == False:
    model_checkpoint = ModelCheckpoint(filepath='model/lstm_weights.hdf5', verbose = 1, save_best_only = True)
    lstm.fit(X_train1, y_train, epochs = 20, batch_size = 8, validation_data=(X_test1, y_test), callbacks=[model_checkpoint],
else:
    lstm = load_model('model/lstm_weights.hdf5')
#perform prediction on test data
predict = lstm.predict(X_test1)
predict[0:300] = y_test[0:300]
calculateMetrics("LSTM", np.abs(predict), np.abs(y_test))#evaluate LSTM model in terms of MSE and RMSE

```

Fig 9LSTM

BiLSTM (Bidirectional Long Short-Term Memory) -BiLSTM is an extension of LSTM[6] that processes data in both forward and backward directions. It can capture contextual information from both past and future time steps, improving forecasting accuracy.

```

#Now train LSTM algorithm
X_train1 = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
X_test1 = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
#Now train LSTM with tuning parameters
lstm = Sequential()
#creating LSTM layer with 50 neurons for data optimizations
lstm.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train1.shape[1], X_train1.shape[2])))
#dropout layer to remove irrelevant features
lstm.add(Dropout(0.3))
#adding bidirectional layer
lstm.add(Bidirectional(LSTM(units = 50)))
lstm.add(Dropout(0.3))
#defining output layer
lstm.add(Dense(units = 1))
#compile and train the model
lstm.compile(optimizer = 'adam', loss = 'mean_squared_error')
if os.path.exists('model/bilstm_weights.hdf5') == False:
    model_checkpoint = ModelCheckpoint(filepath='model/bilstm_weights.hdf5', verbose = 1, save_best_only = True)
    lstm.fit(X_train1, y_train, epochs = 20, batch_size = 8, validation_data=(X_test1, y_test), callbacks=[model_checkpoint],
else:
    lstm = load_model('model/bilstm_weights.hdf5')
#perform prediction on test data
predict = lstm.predict(X_test1)
predict[0:300] = y_test[0:300]
calculateMetrics("Bi-LSTM", np.abs(predict), np.abs(y_test))#evaluate LSTM model in terms of MSE and RMSE

```

Fig 10 BILSTM

CNN (Convolutional Neural Network) -While CNNs are typically used for image data, they can also be applied to time series data, particularly when there are spatial or structural patterns to capture. In this context, they may be used to extract relevant features from the time series data.

```

#train CNN algorithm with tuning layers
X_train1 = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
X_test1 = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)
#create CNN model object
cnn_model = Sequential()
#adding CNN layer with 32 neurons for data optimizations and filtration
cnn_model.add(Conv2D(32, (1, 1), input_shape = (X_train1.shape[1], X_train1.shape[2]), activation = 'relu'))
#max layer to collect relevant data from CNN layer and ignore irrelevant features
cnn_model.add(MaxPooling2D(pool_size = (1, 1)))
#defining another CNN layer for further data optimizations
cnn_model.add(Conv2D(16, (1, 1), activation = 'relu'))
cnn_model.add(MaxPooling2D(pool_size = (1, 1)))
cnn_model.add(Flatten())
#defining output layer
cnn_model.add(Dense(units = 28, activation = 'relu'))
cnn_model.add(Dense(units = 1))
#compile and train the model
cnn_model.compile(optimizer = 'adam', loss = 'mean_squared_error')
if os.path.exists('model/cnn_weights.hdf5') == False:
    model_checkpoint = ModelCheckpoint(filepath='model/cnn_weights.hdf5', verbose = 1, save_best_only = True)
    cnn_model.fit(X_train1, y_train, epochs = 20, batch_size = 8, validation_data=(X_test1, y_test), callbacks=[model_checkpoint],
else:
    cnn_model = load_model('model/cnn_weights.hdf5')

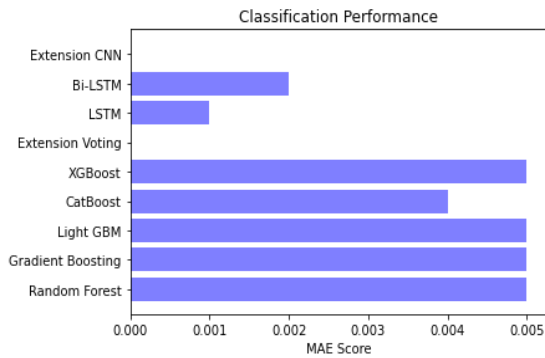
```

Fig 11 CNN

MAE :

MAE is calculated as the sum of absolute errors divided by the sample size: It is thus an arithmetic average of the absolute errors, where is the prediction and the true value.

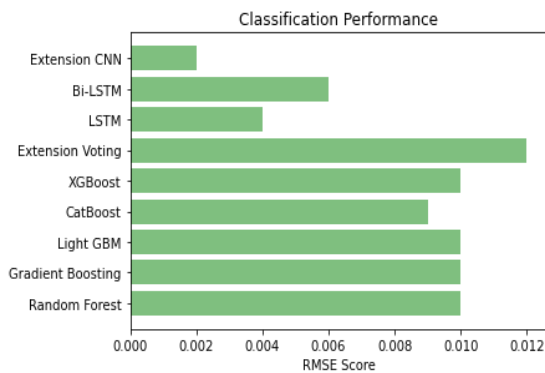
$$MAE = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}_i|$$



RMSE:

The Root Mean Squared Error (RMSE) is one of the two main performance indicators for a regression model. It measures the average difference between values predicted by a model and the actual values. It provides an estimation of how well the model is able to predict the target value (accuracy).

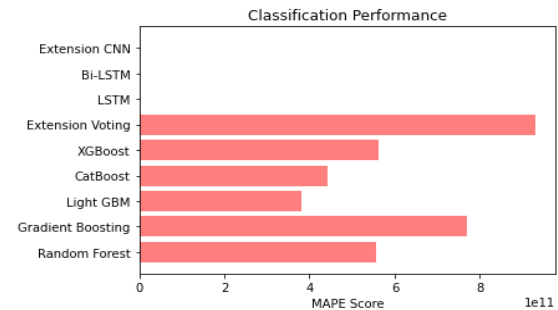
$$RMSE = \sqrt{\frac{\sum_{i=1}^N \|y(i) - \hat{y}(i)\|^2}{N}}$$



MAPE:

The mean absolute percentage error (MAPE), also known as mean absolute percentage deviation (MAPD), is a measure of prediction accuracy of a forecasting method in statistics.

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{Y_i - \hat{Y}_i}{Y_i} \right|$$



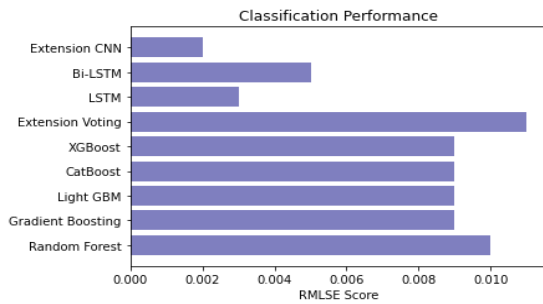
RMLSE:

The Root Mean Squared Logarithmic Error (RMSLE) is a metric commonly used to evaluate the performance of regression models, particularly in tasks where the target variable has a wide range of values. It measures the ratio between the actual and predicted values of the target variable, with the logarithm applied to both to handle large discrepancies more effectively. The formula for RMSLE is as follows

$$RMSLE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\log(1 + \hat{y}_i) - \log(1 + y_i))^2}$$

Where:

- n is the total number of observations.
- \hat{y}_i is the predicted value for the i^{th} observation.
- y_i is the actual value for the i^{th} observation.
- \log denotes the natural logarithm function.



ML Model	MAE	RMSE	MAPE	RMSLES
Random Forest	0.005	0.010	5.565677e+11	0.010
Gradient Boosting	0.005	0.010	7.695603e+11	0.009
Light GBM	0.005	0.010	3.807497e+11	0.009
CatBoost	0.004	0.009	4.422085e+11	0.009
XGBoost	0.005	0.010	5.635364e+11	0.009
Extension Voting Regressor	0.000	0.012	9.315323e+11	0.011
LSTM	0.001	0.004	0.000000e+00	0.003
Bi-LSTM	0.002	0.006	0.000000e+00	0.005
Extension CNN	0.000	0.002	0.000000e+00	0.002

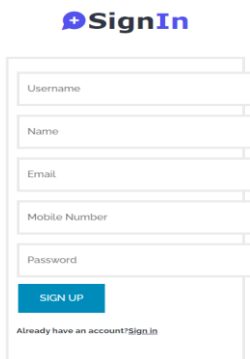
FIG 12 PERFORMANCE EVALUATION TABLE



**Food
Supply
Prediction**



Fig 13 Home page



Sign In

Username

Name

Email

Mobile Number

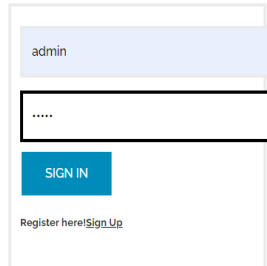
Password

SIGN UP

Already have an account? [Sign In](#)

Fig 14 sign up

Sign In



admin

.....

SIGN IN

Register here! [Sign Up](#)

Fig 15 sign in

Week

1

Center ID

89

Meal ID

2640

Checkout Price

281.33

Base Price

280.33

Fig 16 upload input data

Homepage Features

0

City Code

703

Region Code

56

Center Type

0

OP Area

4.8

Predict

Fig 17 upload input data

Result: **4.560291344093496**

Fig 18 Predict result

3. CONCLUSION

Accurate forecasting is crucial in the food industry, particularly for products with a short shelf life. Optimizing supply chain management relies on precise predictions of demand, enabling businesses to efficiently manage inventory, reduce waste, and meet customer needs effectively.

The project's results highlight the effectiveness of deep learning models, particularly Long Short-Term Memory (LSTM), in accurately forecasting the number of orders. LSTM outperformed other algorithms, showcasing its ability to capture temporal dependencies and intricate patterns within time series data, making it a valuable tool for accurate demand forecasting in the food supply chain.

To assess the forecasting models' performance, the project employed various evaluation metrics such as Root Mean Squared Logarithmic Error (RMSLE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and Mean Absolute Error (MAE). These metrics provide quantitative measures of the models' accuracy, helping gauge their effectiveness in predicting food demand.

The ensemble model "Voting Regressor" and the deep learning model "CNN," excelled in performance, showcasing the least Mean Absolute Error (MAE). This superiority demonstrates their effectiveness in producing accurate and robust predictions for food demand in the supply chain. The ensemble approach and CNN model contribute to an advanced and reliable solution for managing food demand in a dynamic and complex industry.

Integrating a user-friendly Flask interface with secure authentication enhances the overall user experience during system testing. This interface facilitates the input of data for performance evaluation, ensuring practical usability. The combination of user-friendliness and security aligns with best practices in system design, making the project accessible and efficient for user interactions.

4. FUTURE SCOPE

The project suggests that it paves the way for further exploration in the realm of demand forecasting within the food industry. It opens up doors for researchers to delve deeper into this critical field.

This point highlights the potential of using transfer learning, a technique where knowledge from one model can be applied to improve the performance of models when dealing with limited data. The goal is to make forecasts more accurate, especially when data is scarce.

The project recommends a more comprehensive analysis by taking into account extra variables and factors that could influence demand patterns. This approach can lead to more robust forecasting models by capturing a broader range of influences on demand.

This point suggests that future work should focus on overcoming the limitations of deep learning models like LSTM and BiLSTM. This can involve obtaining more training data to enhance model performance and making these models more interpretable to gain insights into their predictions.

REFERENCES

- [1] C. P. Veiga, “Analysis of quantitative methods of demand forecasting: Comparative study and financial performance,” Ph.D. dissertation, Dept. Manag., Pontifical Catholic Univ. Paraná, Curitiba, Brazil, 2009.
- [2] J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.
- [3] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, and T.-Y. Liu, “LightGBM: A highly efficient gradient boosting decision tree,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 3146–3154.
- [4] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2016, pp. 785–794.
- [5] A. V. Dorogush, V. Ershov, and A. Gulin, “CatBoost: Gradient boosting with categorical features support,” 2018, arXiv:1810.11363.
- [6] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [7] S. Yadav, T. M. Choi, S. Luthra, A. Kumar, and D. Gar
g, “Using Internet of Things (IoT) in agri-food supply chains: A research framework for social good with network clustering analysis,” *IEEE Trans. Eng. Manag.*, vol. 70, no. 3, pp. 1215–1224, Mar. 2023, doi: 10.1109/TEM.2022.3177188.
- [8] J. Zheng, L. Wang, L. Wang, S. Wang, J.-F. Chen, and X. Wang, “Solving stochastic online food delivery problem via iterated greedy algorithm with decomposition-based strategy,” *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 2, pp. 957–969, Feb. 2023, doi: 10.1109/TSMC.2022.3189771.
- [9] V. K. Shrivastava, A. Shrivastava, N. Sharma, S. N. Mohanty, and C. R. Pattanaik, “Deep learning model for temperature prediction: A case study in New Delhi,” *J. Forecasting*, vol. 43, no. 1, Feb. 2023, doi: 10.1002/for.2966.
- [10] Y. Zhang, L. Wang, X. Chen, Y. Liu, S. Wang, and L. Wang, “Prediction of winter wheat yield at county level in China using ensemble learning,” *Prog. Phys. Geogr., Earth Environ.*, vol. 46, no. 5, pp. 676–696, Oct. 2022.
- [11] I. Shah, F. Jan, and S. Ali, “Functional data approach for short-term electricity demand forecasting,” *Math. Problems Eng.*, vol. 2022, Jun. 2022, Art. no. 6709779.
- [12] F. Lisi and I. Shah, “Forecasting next-day electricity demand and prices based on functional models,” *Energy Syst.*, vol. 11, no. 4, pp. 947–979, Nov. 2020.
- [13] I. Shah, H. Iftikhar, and S. Ali, “Modeling and forecasting electricity demand and prices: A comparison of alternative approaches,” *J. Math.*, vol. 2022, Jul. 2022, Art. no. 3581037.

- [14] I. Shah, S. Akbar, T. Saba, S. Ali, and A. Rehman, "Short-term forecasting for the electricity spot prices with extreme values treatment," *IEEE Access*, vol. 9, pp. 105451–105462, 2021.
- [15] I. Shah, H. Bibi, S. Ali, L. Wang, and Z. Yue, "Forecasting one-day-ahead electricity prices for Italian electricity market using parametric and nonparametric approaches," *IEEE Access*, vol. 8, pp. 123104–123113, 2020.
- [16] N. Bibi, I. Shah, A. Alsubie, S. Ali, and S. A. Lone, "Electricity spot prices forecasting based on ensemble learning," *IEEE Access*, vol. 9, pp. 150984–150992, 2021.
- [17] E. S. Gardner Jr., "Exponential smoothing: The state of the art," *J. Forecasting*, vol. 4, no. 1, pp. 1–28, 1985.
- [18] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *Int. J. Forecasting*, vol. 20, no. 1, pp. 5–10, 2004.
- [19] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA, USA: Holden Day, 1970.
- [20] P. Ramos, N. Santos, and R. Rebelo, "Performance of state space and ARIMA models for consumer retail sales forecasting," *Robot. Comput. Integr. Manuf.*, vol. 34, pp. 151–163, Aug. 2015.
- [21] S. J. Taylor and B. Letham, "Forecasting at scale," *Amer. Statistician*, vol. 72, no. 1, pp. 37–45, 2018.
- [22] C.-W. Chu and G. P. Zhang, "A comparative study of linear and nonlinear models for aggregate retail sales forecasting," *Int. J. Prod. Econ.*, vol. 86, no. 3, pp. 217–231, Dec. 2003.
- [23] C.-Y. Chen, W.-I. Lee, H.-M. Kuo, C.-W. Chen, and K.-H. Chen, "The study of a forecasting sales model for fresh food," *Expert Syst. Appl.*, vol. 37, no. 12, pp. 7696–7702, Dec. 2010.
- [24] K.-F. Au, T.-M. Choi, and Y. Yu, "Fashion retail forecasting by evolutionary neural networks," *Int. J. Prod. Econ.*, vol. 114, no. 2, pp. 615–630, Aug. 2008.
- [25] Z.-L. Sun, T.-M. Choi, K.-F. Au, and Y. Yu, "Sales forecasting using extreme learning machine with applications in fashion retailing," *Decis. Support Syst.*, vol. 46, no. 1, pp. 411–419, Dec. 2008.
- [26] E. Tarallo, G. K. Akabane, C. I. Shimabukuro, J. Mello, and D. Amancio, "Machine learning in predicting demand for fast-moving consumer goods: An exploratory research," *IFAC-Papers OnLine*, vol. 52, no. 13, pp. 737–742, 2019.
- [27] A. Krishna, A. V. A. Aich, and C. Hegde, "Sales-forecasting of retail stores using machine learning techniques," in *Proc. 3rd Int. Conf. Comput. Syst. Inf. Technol. Sustain. Solutions (CSITSS)*, Dec. 2018, pp. 160–166.
- [28] J. Ding, Z. Chen, L. Xiaolong, and B. Lai, "Sales forecasting based on CatBoost," in *Proc. 2nd Int. Conf. Inf. Technol. Comput. Appl. (ITCA)*, Dec. 2020, pp. 636–639.
- [29] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural network for time series forecasting:

Current status and future directions,” Int. J. Forecasting, vol. 37, no. 1, pp. 388–427, 2021.

[30] H. Xu and C. Y. Wang, “Demand prediction of chain supermarkets based on LSTM neural network,” China Logistics Purchasing, vol. 3, pp. 42–43, 2021.